Labelling Scheduler: A Flexible **Labelling-based Jointly Scheduling** Approach for **Big Data Analysis**

Authors： Xin Li , Zhuzhong Qian, Jianjun Qiu , Xiaolin Qin , Jie Wu

Nanjing University of Aeronautics and Astronautics

25th IEEE International Conference on Parallel and Distributed Systems

Construction

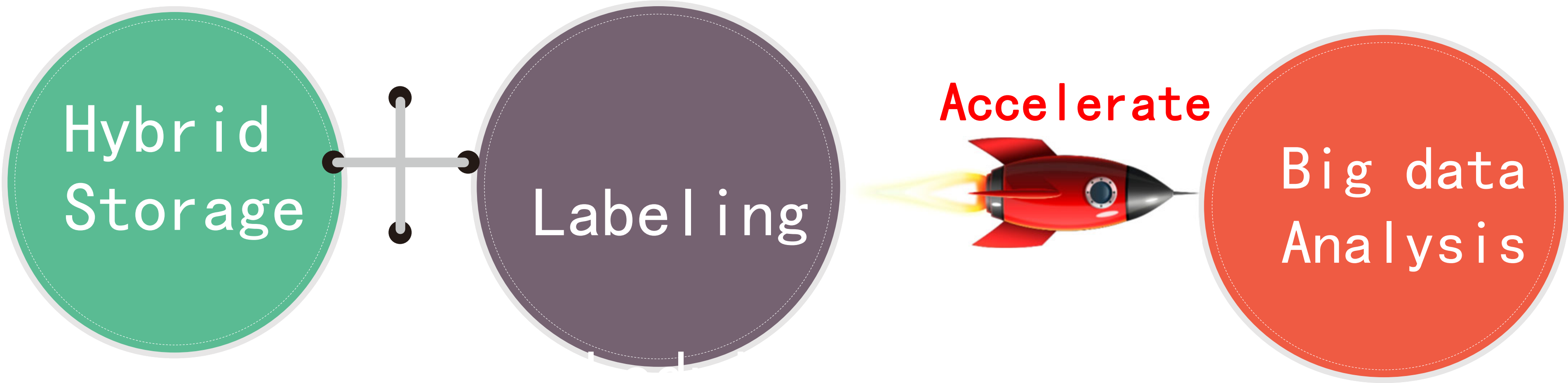
Ⅰ. Introduction

Ⅰ. Introduction -- A. **Motivation**

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Read performance of disk is rather poor → Non-Volatile Memory (NVM)

More readings and few writings → Motivates us to utilize NVM



Ⅰ. Introduction -- B. **Scenario**

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

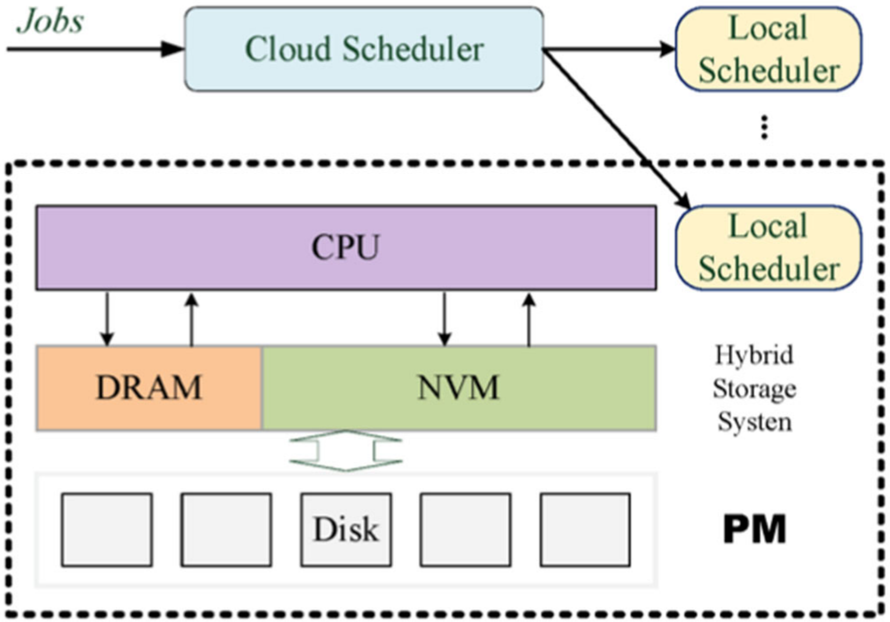


Fig. 1. System Scenario

Cloud data center with a hybrid storage system, which includes DRAM, NVM, and Disk.

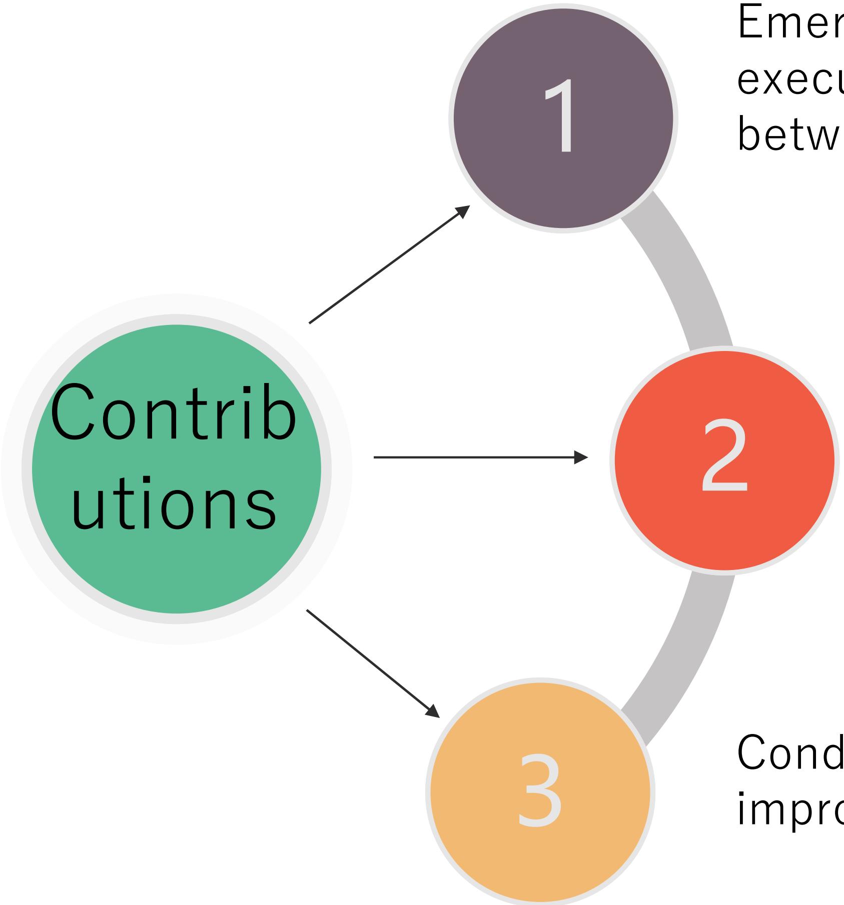
Location of data object affects the job execution time significantly.

It is preferable to place data in NVM.

So,there must be some data replacement between NVM and disk to achieve better performance.

Ⅰ. Introduction -- C. **Contributions**

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS



1. Emerge NVM to speed up the big data analysis job execution, and establish the data replacement model between disk and NVM.

2. Take data features into account → labeling method → *labeling scheduler*.

3. Conduct extensive simulations → significant performance improvement.

Ⅱ. Problem Statement

Ⅱ. Problem Statement -- A Preliminaries

$$J = \langle \Gamma_a, \mathcal{I}, \theta, \chi, \omega, \mathcal{S} \rangle$$

- Γ_a is the job arrival time to the cloud system;
- θ is the required duration to return the analysis result;
- χ is the real duration between job arrival and completion;
- ω is the full utility for finishing the job in time;
- $\mathcal{S}$ is the set of tasks contained in the job. For the items in $\mathcal{S}$, we call them the job's children tasks and the job is known as their parent job. The job is completed only when all children tasks are finished;
- $\mathcal{I}$ is used to identify the job is an interactive job or batch job. For the job J_k, we have

$$\mathcal{I}(J_k) = \begin{cases} 1, & J_k \text{ is an interactive job;} \\ 0, & \text{otherwise.} \end{cases}$$

1)The physical computing resource is split into m computing slots, or virtual machines (VMs).

2)The NVM is also split into n storage slots.

3)For each task execution, it will occupy one VM and read data from one slot in NVM or disk.

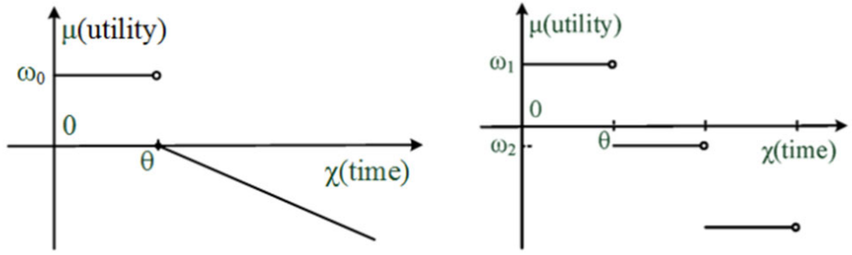
Ⅱ. Problem Statement -- B

Utility Functions

Resource is limited, and we need to select some jobs with high priority to occupy the computing resource slots.

Therefore, there may be some jobs that cannot be completed before the deadline, leading a negative utility.

$$\mu(J_i) = \omega_1 - (\omega_1 - \omega_2)\lfloor \frac{\chi}{\theta} \rfloor.$$



(a) Interactive Job (b) Batch Job

Fig. 2. Utility Functions

Ⅱ. Problem Statement -- C

Job Execution Time

execution time under the NVM-case

task execution time

$$\Gamma_e(A_i) = \begin{cases} \Gamma_N(A_i), & loc(d(A_i)) = 0; \\ \Gamma_N(A_i) + \Gamma_r, & \text{otherwise.} \end{cases}$$

$$= \Gamma_N(A_i) + loc(d(A_i)) \cdot \Gamma_r.$$

data migration time

$$loc(D_i) = \begin{cases} 0, & D_i \text{ is stored in the NVM;} \\ 1, & \text{otherwise.} \end{cases}$$

job execution time

$$\Gamma_e(J_k) = \sum_{A_i \in J_k} \Gamma_e(A_i).$$

Ⅱ. Problem Statement -- D

Jointly Scheduling Problem

$$max. \quad \sum_{i=1}^{\kappa} \mu(J_i).$$

Theorem 1: The jointly scheduling problem is NP-hard.

Features of data are critical for data placement. This motivates us to explore the labeling method, which will help us to conduct data selection during data replacement.

III. Labelling System

Ⅲ. Labelling System -- A

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Label: hotness

$$hotness(D_i) += H \cdot K, \quad \longrightarrow \quad hotness(D_i) += \sum_{J_k \in U(D_i)} H1 + H2 \cdot \mathcal{I}(J_k)$$

At the beginning of each time-slot, the value of hotness decreases by 1 （avoid value infinitely）．

hotness(Di)+△H

Ⅲ. Labelling System -- B

Label: class

$$
class(D_i) = \begin{cases}
0, & \exists A_k,\ d(A_k) = D_i,\ A_k \text{ is running;} \\
1, & \exists J_k \in \mathcal{Q},\ I(J_k) = 1 \wedge D_i \in d(J_k); \\
2, & \exists J_k \in \mathcal{Q},\ D_i \in d(J_k), \text{ and} \\
 & J_k \text{ is the actress batch job;} \\
3, & D_i \text{ is read by interactive job before;} \\
4, & \exists J_k \in \mathcal{Q},\ D_i \in d(J_k), \text{ and} \\
 & J_k \text{ is an audience batch job;} \\
5, & \text{otherwise.}
\end{cases}
$$

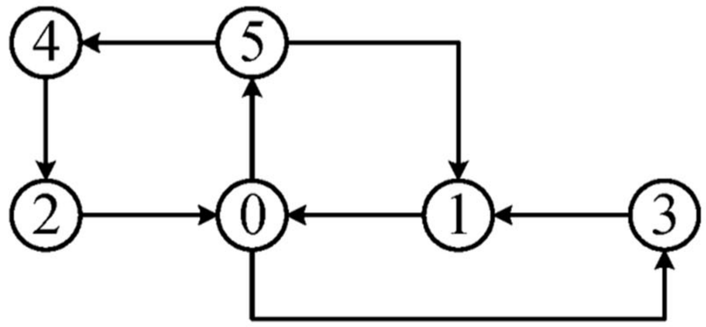


Fig. 3. Class State Transformation.

IV. Labelling Scheduler

Ⅳ. Labelling Scheduler -- A **Basis Ideas**

The basic ideas for the 3 components are summarized as：

- *Job Admission*. For the new candidate job, compare the workload and usable computing resource before the LST of the job.
- *Job Scheduling*. Once some computing slot is available, take LST of the jobs in queue Q as the major concern for job/task selection.
- *Data Replacement*. Take both the hotness and class into account comprehensively for data selection when the data bus is free.

Ⅳ. Labelling Scheduler -- B

Load-aware Job Admission

Algorithm 1 $admission(J_i)$

Input: J_i: the arrived job, $\mathcal{Q}$: the set of jobs of the queue.

for *each* $A_k \in \mathcal{S}(J_i)$ **do**
    $hotness(d(A_k)) += H_1 + H_2 \cdot \mathcal{I}(J_i)$;
$load \leftarrow 0$;
$LST(J_i) \leftarrow \Gamma_a(J_i) + \theta(J_i) - exe(J_i)$;
for *each* $J_k \in Q$ **do**
    if $deadline(J_k) < LST(J_i)$ **then**
        $load += expLoad(J_k)$;
if $\lceil \frac{load}{m} \rceil \leq LST(J_i)$ **then**
    accept J_k;
    update data label $class$;

$$jobLoad(J_k) = \sum_{A_i \in J_k} (\Gamma_N(A_i) + loc(d(A_i)) \cdot p \cdot \Gamma_r),$$

Ⅳ. Labelling Scheduler -- C
LST-based Job Scheduling

Algorithm 2 LST: $jobScheduling(VM_i)$

Input: Q: the set of jobs of the queue.

$LST \leftarrow \infty$, $\gamma \leftarrow \infty$;
$J \leftarrow NULL$, $A \leftarrow NULL$;
for *each* $J_k \in Q$ **do**
    if $LST < LST(J_k)$ **then**
        $LST \leftarrow LST(J_k)$;
        $J \leftarrow J_k$;
for *each* $A_i \in J$ **do**
    if $\gamma > \Gamma_e(A_i)$ **then**
        $\gamma \leftarrow \Gamma_e(A_i)$;
        $A \leftarrow A_i$;
assign VM_i to task A;
$class(d(A)) \leftarrow 0$;

Ⅳ. Labelling Scheduler -- D
LST-based Job Scheduling

- Q1: Is it necessary to migrate a new data to NVM?
- Q2: Which data should be migrated to NVM?
- Q3: Which data in NVM have to be replaced?

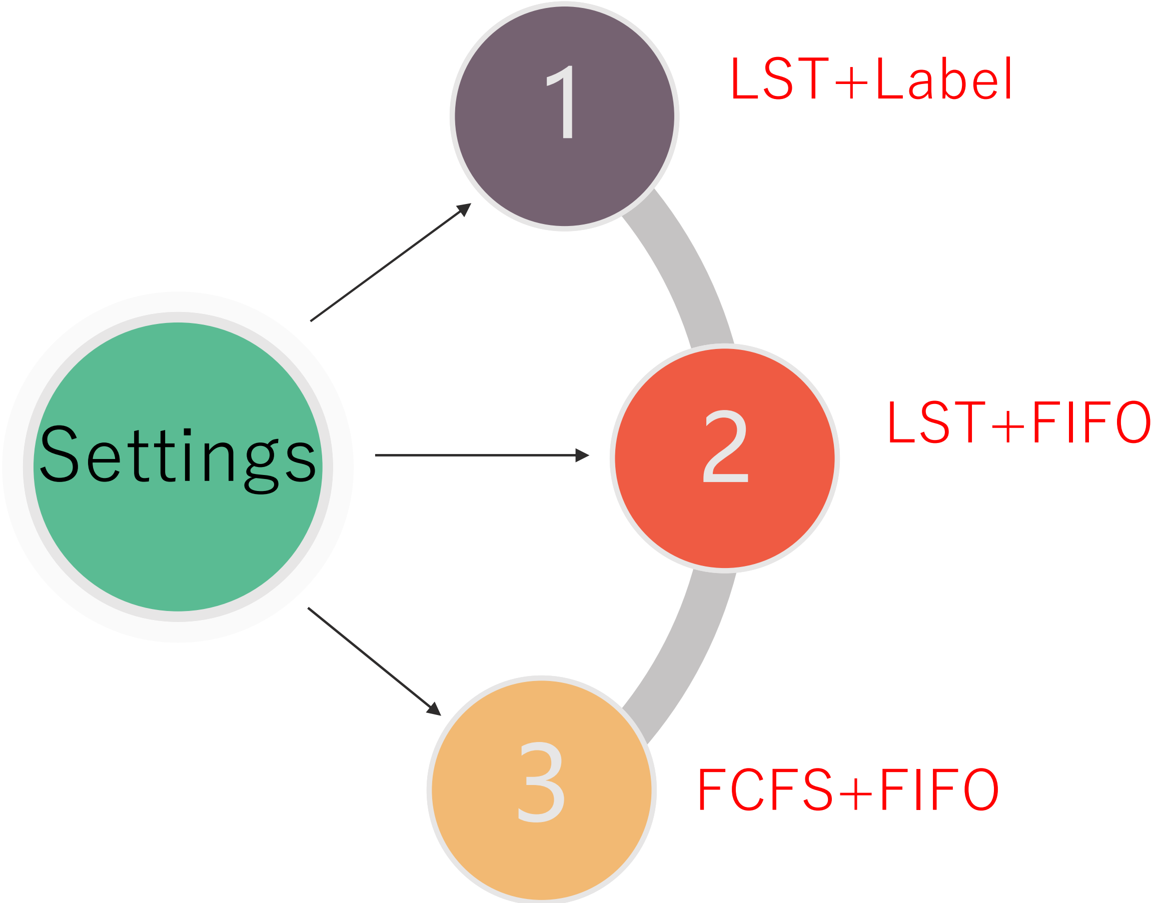
Algorithm 3 Label: $dataReplacment()$

Input: Mi and Kj: the data subsets,
δ: threshold (constant number)

$M \leftarrow merge(M0, M1, M2, M3, M4, M5)$;
$K \leftarrow merge(K0, K1, K2, K3, K4, K5)$;
if $M \neq \emptyset$ **then**
  $D_C \leftarrow lastData(M)$;
if $K \neq \emptyset$ **then**
  $D_H \leftarrow firstData(K)$;
  if $class(D_H) = 2 \&\& |M2| \leq \delta$ **then**
    if $K3 \neq \emptyset$ **then**
      $D_H \leftarrow firstData(K3)$;
if $class(D_H) < class(D_C)$ **then**
  $migrate(D_H, D_C)$;
else if $class(D_H) = 3 \;\&\&\; class(D_C) = 3$ **then**
  if $hotness(D_H) + \Delta H(D_H) > hotness(D_C) + \Delta H(D_C)$ **then**
    $migrate(D_H, D_C)$;

V. Performance Evaluation

VI. Performance Evaluation -- A
Simulation Settings

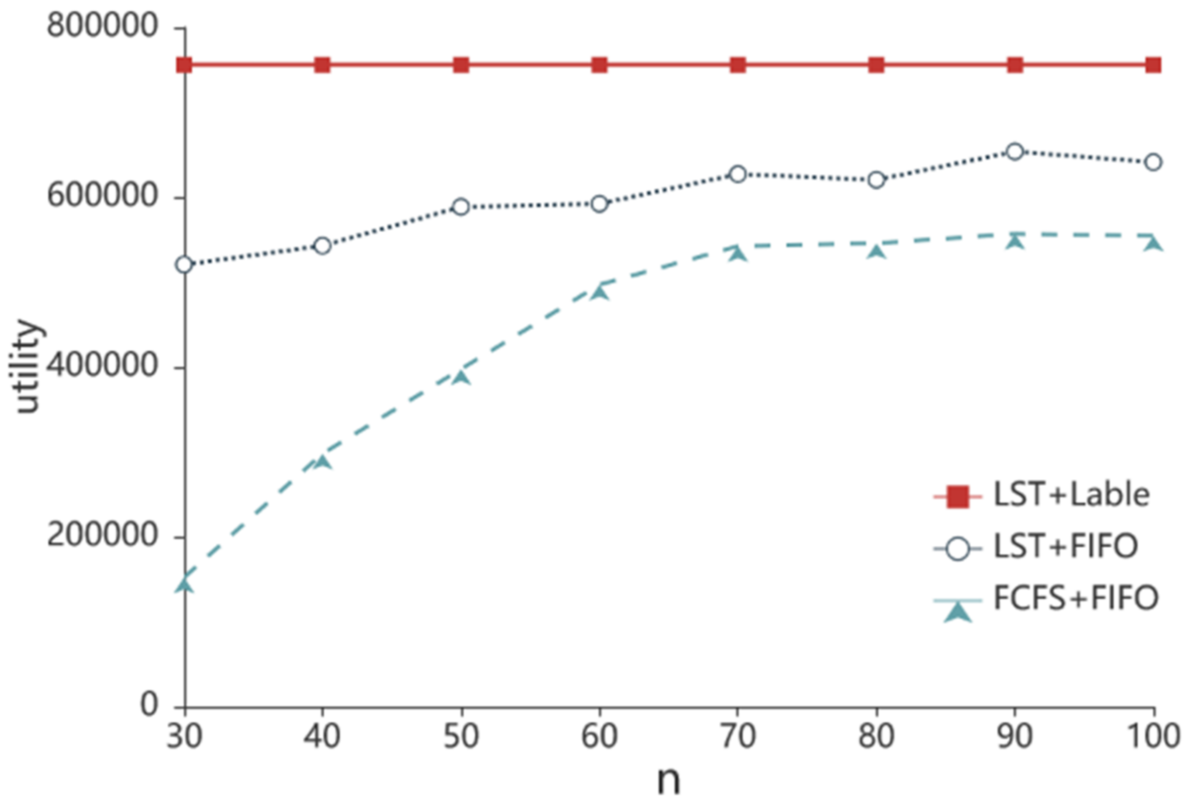


For the physical resource, the computing slot number m and the NVM slot number n .

Let m = 64, n = 60 be the typical setting, and change n from 30 to 100.

VI. Performance Evaluation -- B

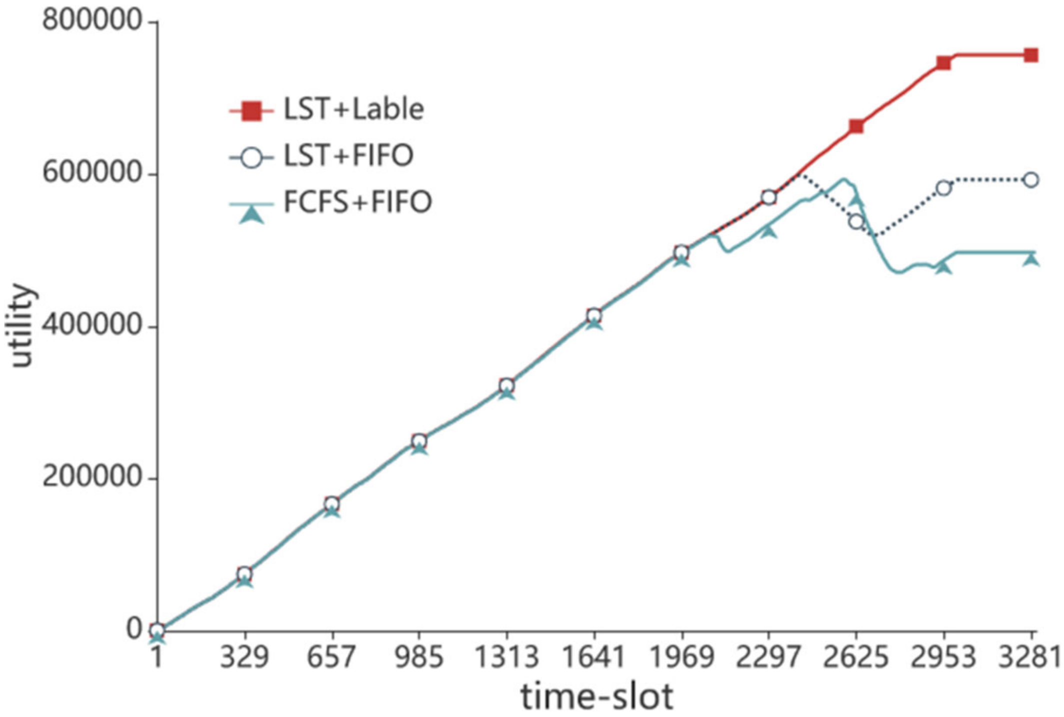
Result Analysis

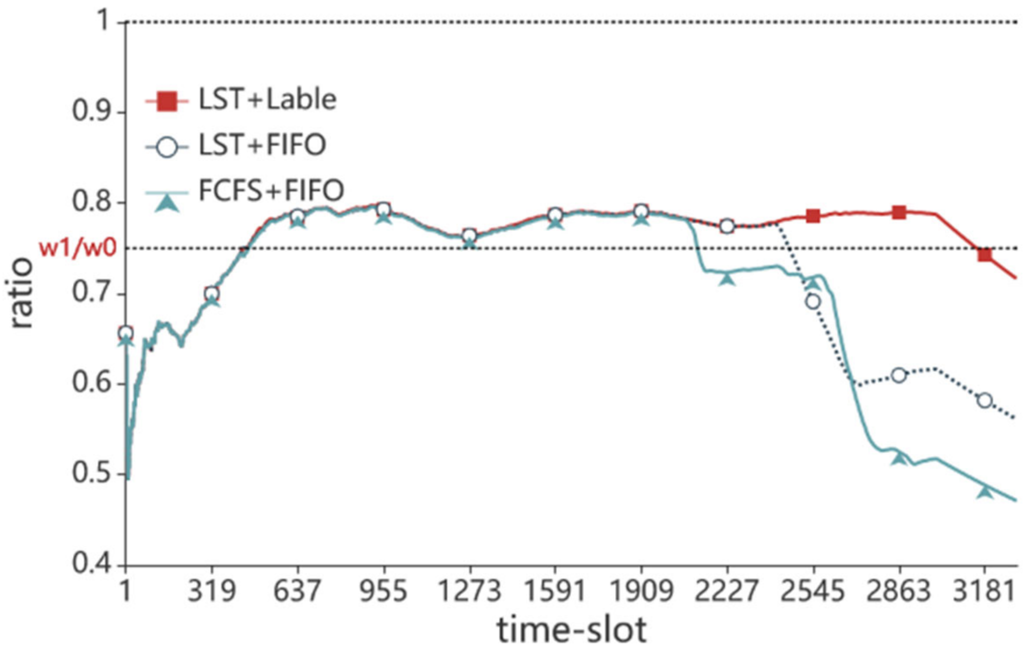


Utility: m = 64

VI. Performance Evaluation -- B

Result Analysis



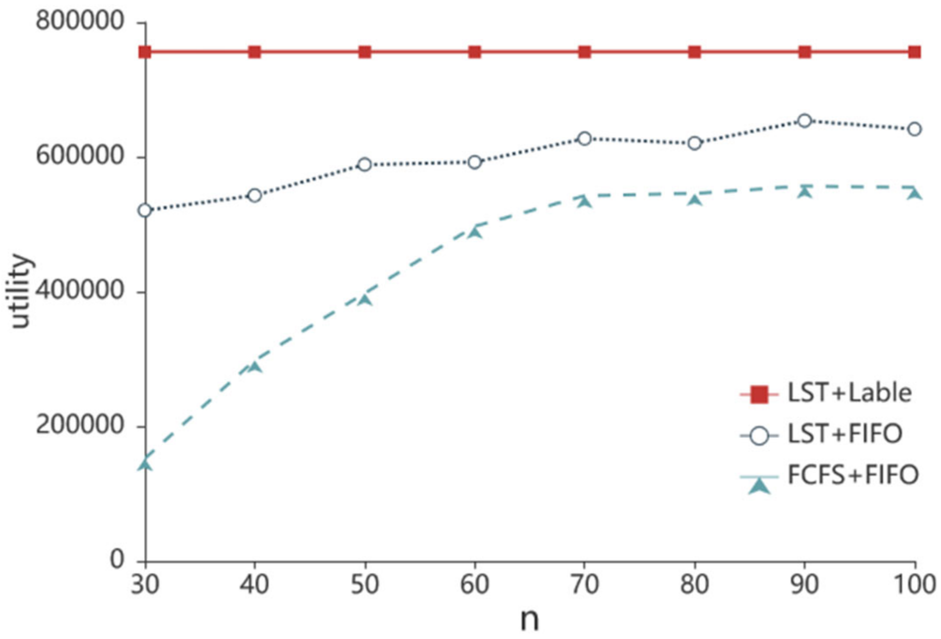


Details of utilities: m = 64, n = 60

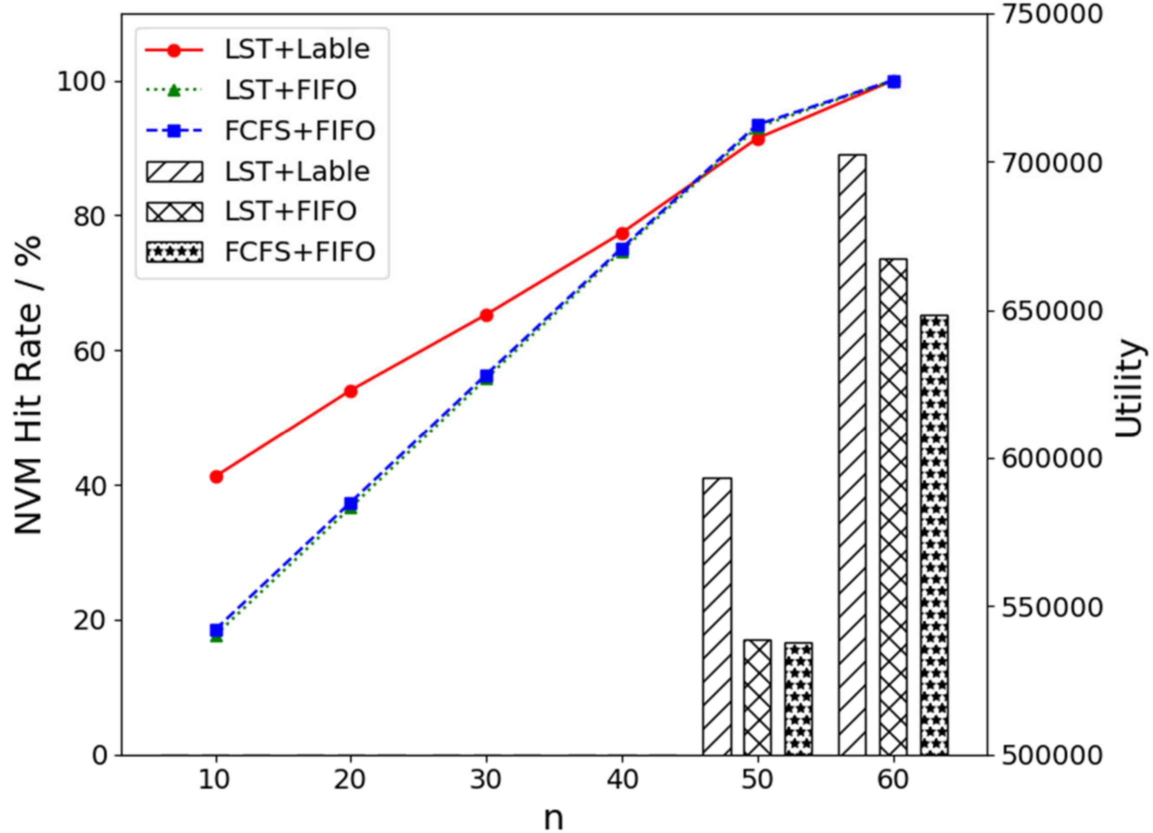
VI. Performance Evaluation -- B

NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Result Analysis--NVM Hit Rates Analysis



NVM hit rate when m=64



NVM hit rate when m=56

VI. Conclusion

VI. Conclusion

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Investigate the joint job and data scheduling problem for utility maximization in cloud data center with a hybrid storage system.

(1) Labeling system.

(2) A flexible labeling-based approach for joint job and data scheduling.

(3) Extensive simulations show that the proposed ***labeling scheduler*** has significant performance.

Thanks To All of You !

4th,Dec,2019